REMARKS

In response to the Office Action mailed April 13, 2005, Applicants respectfully request reconsideration. Claims 1-18 were previously pending in this application and claims 1, 10, and 16 are amended herein. The application as presented is believed to be in condition for allowance.

The Office Action rejected claims 1-5, 8-10, 12, and 14 under 35 U.S.C. §103(a) as purportedly being obvious over Song (5,546,599) in view of Grochowski (6,353,883). The Office Action further rejected claim 16 under 35 U.S.C. §103(a) as purportedly being obvious over the combination of Song, Grochowski, and Hennessy (Hennessy and Patterson, Computer Architecture – A Quantitative Approach, 2nd Edition, 1996). The Office Action further rejected claims 6, 7, 11, 13, 15, 17, and 18 under 35 U.S.C. §103(a) as purportedly being obvious over Song and Grochowski in combination with various other references. Each of these rejections is respectfully traversed.

Claim Objections

The Office Action objected to claims 1, 10, and 16, asserting that certain phrases recited therein lack antecedent basis. Applicants have amended these claims to address this issue. Accordingly, withdrawal of these objections is respectfully requested.

The Combination of Song and Grochowski

Applicants maintain that the combination of Song and Grochowski is improper. The Office Action asserts, "[a] person of ordinary skill in the art would have recognized that by implementing predicate prediction within Song, a) conditional branches could be eliminated, thereby reducing the amount of instructions required in the instruction set, and b) predicated instructions would be speculatively executed (i.e., executed before the corresponding guard is resolved), thereby maximizing throughput by executing predicated instructions as soon as possible. *See* Office Action, page 5, lines 15-20.

Applicants respectfully disagree with this reasoning. While it is true that some predicated processors may not require the use of branch instructions and thus the number of instructions in

the instruction set is reduced, this benefit may be offset by the fact that every other instruction includes additional bits for the predicate field that are not required in non-predicated processors. Thus, while there may be fewer types of instructions in a predicated processor (i.e., because branch instructions are not used) the total amount of code in a program may actually be increased due to the fact that additional bits for the predicate field must be included in every other instruction.

Applicants further disagree that a predicated processor maximizes throughput by executing predicated instructions as soon as possible. In predicated processors, many instructions may make it most of the way through the execution pipeline whether they are to be executed or not. Thus, for instructions whose predicate is eventually resolved as false, many execution cycles may have been wasted processing the instruction in the pipeline, even though the instruction will never be executed. Applicants respectfully disagree that this maximizes processor throughput.

Further, the processor of Song processes multiple instructions simultaneously and executes instructions out of order, relative to their programmed sequence. That is, Song discusses that in some situations it may be necessary to delay execution of certain instructions because the operands of these instructions may depend on the result of previous instructions that have not yet completed execution (Col. 5, lines 40-54). Thus, it may not be preferential or even possible in the system of Song to execution instructions "as soon as possible," as the instruction should be delayed to wait for the result of another instruction.

Moreover, because the processor 10 of Song is an out-of-order processor that executes multiple instructions at the same time using multiple execution units and relies on a sequencer unit 18 to determine the order and timing in which instructions should be sent to the various execution units, it would be difficult, if not impossible, to incorporate predicates into the system of Song without upsetting the timing and synchronization that is required in executing multiple instructions at the same time, using different execution units. There is certainly no teaching in either reference that the processor of Song could be modified to perform predicated execution or why one would have been motivated to do so when the processor of Song, in connection with the

sequencer unit, are already provided to optimize execution of instructions and avoid wasted execution cycles.

If the rejection is to be maintained, Applicants respectfully request that the Examiner cite a reference explaining how predicated instructions could be incorporated into the out-of-order processor disclosed by Song and why predicated execution would be beneficial in the system of Song.

Claim 1 Patentably Distinguishes Over the Combination of Song and Grochowski

Even if Song and Grochowski were combined in the manner asserted in the Office Action, claim 1 patentably distinguishes over any such combination.

Claim 1 is directed to a computer system for executing predicated instructions wherein each instruction includes a guard, the value of which determines whether or not that instruction is executed, the computer system comprising: a fetch unit for fetching instructions to be executed; a decode unit for decoding said instructions; at least one pipelined execution unit for executing decoded instructions and being associated with a guard register file holding values of the guards to allow resolution of the guards to be made; and an emulation unit including control circuitry which cooperates with the decode unit to selectively control the decode unit to implement a precise watch or a non-precise watch on detection of a breakpoint caused by a fetched instruction having a specified program count or a fetched instruction having a specified opcode, wherein according to a precise watch, the fetched instruction causing the breakpoint and subsequent instructions are not allowed to enter the execution unit and, according to a non-precise watch, the instruction causing the breakpoint and subsequent instructions are permitted to be supplied from the decode unit to the at least one execution unit while guard resolution in said at least one execution pipeline is awaited.

The Office Action asserts that Song discloses, at column 9, line 8 – column 10, line 7, an emulation unit including control circuitry which cooperates with the decode unit to selectively control the decode unit to implement a precise watch or non-precise watch on detection of a breakpoint caused by an instruction having a specified program count or an instruction having a specified opcode, wherein according to a non-precise watch, the instruction causing the

breakpoint and subsequent instructions are not allowed to enter the execution unit. Applicant respectfully disagrees.

Initially, to the extent that Song discusses preventing execution of breakpoint instructions at all, this function is not performed by an emulation unit including control circuitry which cooperates with the decode unit, as required by claim 1. There is no teaching or suggestion in either Song or Grochowski that this function can be performed by an emulation unit (i.e., a unit used for diagnostic or debugging purposes). *See* Applicants' specification, page 1, lines 17-24.

As disclosed in Applicants' specification, using an emulation unit to perform this function addresses a problem that arises in debugging predicated processors. Specifically, on detection of a breakpoint an emulation unit may take over control of the processor for debugging purposes. However, in a predicated processor, if the breakpoint is caused by an instruction whose guard value eventually resolves as false (i.e., an instruction that would never have executed), then the emulation unit taking over control of the processor as a result of that instruction may cause unnecessary intrusion the processor's execution. *See* Applicants' specification, page 1, line 20 – page 2, line 8.

This problem is addressed by providing an emulation unit that operates in either a precise watch mode or non-precise watch mode. In the precise watch mode, upon detection of a breakpoint instruction, subsequent instructions are not permitted to enter the execution pipeline paste the decode/dispatch stage until the guard value of the breakpoint instruction is resolved. That is, the emulation unit causes the decode unit to issue a request for guard resolution. If the guard value is false, then the emulation unit may allow the execution pipeline to continue to normally. If the guard value is true, then the emulation unit may take over operation of the processor for debug purposes. This approach is advantageous because the state of the registers and memory of the processor when the emulator takes over control is the same as the state just before the breakpoint. However, the performance of the processor has been degraded, while waiting for the guard value of the breakpoint instruction to be resolved. *See* Applicants' specification, page 3, lines 22-30 and page 10, lines 8-21.

In the non-precise watch mode, when the emulation unit detects a breakpoint instruction, subsequent instructions are allowed to continue through the execution pipeline normally.

Eventually, the guard value of the breakpoint instruction will be resolved. If the guard value is resolved as true, then the emulation unit may take over operation of the processor for debugging purposes. However, by the time the guard value is resolved, other instructions may have entered the pipeline. This is disadvantageous because the state of the processor at the time of debugging may be slightly different than its state at the time the breakpoint instruction was detected, making debugging slightly more complex. Nevertheless, this approach avoids halting the pipeline on detection of breakpoint instruction to wait for guard values to be resolved, as in the precise watch mode. *See* Applicants' specification, page 10, line 22 – page 11, line 10.

The foregoing summary is provided merely to assist the Examiner in appreciating various aspects of the present invention. The summary may not apply to each of the independent claims, and the language of the independent claims may differ in material respects from the summary provided. The Examiner is requested to give a careful consideration to the language of each of the independent claims and to address each on its own merits, without relying on the summary provided above. Applicants do not rely on the summary to distinguish any of the claims of the present invention over the prior art, but rather, rely only upon the arguments provided below.

Neither Song nor Grochowski teaches or suggests the use of an emulation unit, as both these references are entirely unrelated to debugging. Moreover, neither Song nor Grochowski discloses a precise watch in which "the fetched instruction causing the breakpoint and subsequent instructions are not allowed to enter the execution unit" and a non-precise watch in which "the instruction causing the breakpoint and subsequent instructions are permitted to be supplied from the decode unit to the at least one execution unit," as required by claim 1.

The Office Action asserts that Song teaches this limitation at column 9, line 48 - column 10, line 7 and column 22, lines 16-29. Applicants respectfully disagree. The cited paragraph of Song teaches that for any instruction that causes an exception, the instruction causing the exception is not executed. Song does not teach or suggest that subsequent instructions are not allowed to enter the execution unit. The Office Action asserts that, "in column 10, lines 12-16, it is disclosed that the breakpoint/exception is processed after all instructions preceding the breakpoint instruction are complete. As a result, it should be realized that instructions subsequent to the breakpoint are not executed (don't enter the execution unit) until the exception

is processed." Applicants respectfully disagree. The processor of Song is capable of execution multiple instructions simultaneously using different execution units. Nowhere does Song teach or suggest that an exception causes the pipeline to be halted (with respect to instructions not yet in the pipeline) until the exception is processed. The Office Action asserts that because the exception is not processed until all preceding instructions have completed execution, no new instructions can enter the pipeline until the exception is processed. This is not taught anywhere in Song and is simply not the case.

In view of the foregoing, claim 1 patentably distinguishes over Song and Grochowski, whether taken alone or in combination. Accordingly, it is respectfully requested that rejection of claim 1 under 35 U.S.C. §103(a) be withdrawn.

Claims 2-9 depend from claim 1 and are patentable for at least the same reasons. Accordingly, it is respectfully requested that the rejection of claims 2-9 be withdrawn.

Claim 10 Patentably Distinguishes Over the Combination of Song and Grochowski

Claim 10 is directed to a method of debugging an on-chip processor which is arranged to execute predicated instructions wherein each instruction includes a guard, the value of which determines whether or not that instruction is executed, the method comprising: fetching instructions to be executed; decoding said instructions; executing decoded instructions, said executing step including resolving values of the guards of the instructions; and detecting instructions which have a debug effect based on a program count or an opcode of the instructions and acting on said instructions in dependence on whether the processor is in a precise watch mode or a non-precise watch mode wherein, according to a precise watch mode, an instruction having a debug effect and subsequent instructions are not executed and, according to a non-precise watch mode, the instruction and subsequent instructions are supplied and executed normally while guard resolution is awaited.

As should be clear from the discussion above in, neither Song nor Grochowski teaches or suggests, "detecting instructions which have a debug effect based on a program count or an opcode of the instructions and acting on said instructions in dependence on whether the processor is in a precise watch mode or a non-precise watch mode wherein, according to a

precise watch mode, an instruction having a debug effect and subsequent instructions are not executed and, according to a non-precise watch mode, the instruction and subsequent instructions are supplied and executed normally while guard resolution is awaited."

Claim 10 patentably distinguishes over Song and Grochowski, whether taken alone or in combination. Accordingly, it is respectfully requested that rejection of claim 10 under 35 U.S.C. §103(a) be withdrawn.

Claims 11-15 depend from claim 10 and are patentable for at least the same reasons. Accordingly, it is respectfully requested that the rejection of claims 11-15 be withdrawn.

Claim 16 Patentably Distinguishes Over the Combination of Song and Grochowski and Hennessy

Claim 16, as amended, is directed to a computer system for executing predicated instructions wherein each instruction includes a guard, the value of which determines whether or not that instruction is executed, the computer system comprising: a fetch unit for fetching instructions to be executed; a decode unit for decoding said instructions; at least one pipelined execution unit for executing decoded instructions and being associated with a guard register file holding values of the guards to allow resolution of the guards to be made; and an emulation unit including control circuitry which cooperates with the decode unit to selectively control the decode unit to implement a precise watch or a non-precise watch on detection of a breakpoint caused by a fetched instruction having a specified program count or a fetched instruction having a specified opcode, wherein according to a precise watch, the fetched instruction causing the breakpoint and subsequent instructions are not allowed to enter the execution unit and, according to a non-precise watch, the fetched instruction causing the breakpoint and subsequent instructions are permitted to be supplied from the decode unit to the at least one execution unit while guard resolution in said at least one execution pipeline is awaited, wherein, when the emulation unit is in the precise watch mode, it is operable to issue a request to the execution pipeline for guard resolution, the guard resolution being transmitted to the control circuitry of the emulation unit which is responsive thereto to control operation of the decode unit.

As should be clear from the discussion above, neither Song, Grochowski, or Hennessy teaches or suggests, "an emulation unit including control circuitry which cooperates with the decode unit to selectively control the decode unit to implement a precise watch or a non-precise watch on detection of a breakpoint caused by a fetched instruction having a specified program count or a fetched instruction having a specified opcode, wherein according to a precise watch, the fetched instruction causing the breakpoint and subsequent instructions are not allowed to enter the execution unit and, according to a non-precise watch, the fetched instruction causing the breakpoint and subsequent instructions are permitted to be supplied from the decode unit to the at least one execution unit while guard resolution in said at least one execution pipeline is awaited, wherein, when the emulation unit is in the precise watch mode, it is operable to issue a request to the execution pipeline for guard resolution, the guard resolution being transmitted to the control circuitry of the emulation unit which is responsive thereto to control operation of the decode unit."

Thus, claim 16 patentably distinguishes over Song, Grochowski, and Hennessy, whether taken alone or in combination. Accordingly, it is respectfully requested that rejection of claim 16 under 35 U.S.C. §103(a) be withdrawn.

Claims 17 and 18 depend from claim 16 and are patentable for at least the same reasons. Accordingly, it is respectfully requested that the rejection of claims 17 and 18 be withdrawn.

CONCLUSION

In view of the foregoing amendments and remarks, this application should now be in condition for allowance. A notice to this effect is respectfully requested. If the Examiner believes, after this amendment, that the application is not in condition for allowance, the Examiner is requested to call the Applicant's attorney at the telephone number listed below.

If this response is not considered timely filed and if a request for an extension of time is otherwise absent, Applicant hereby requests any necessary extension of time. If there is a fee occasioned by this response, including an extension fee, that is not covered by an enclosed check, please charge any deficiency to Deposit Account No. 23/2825.

Respectfully submitted,

Andrew Cofler et al.

By: _____

James H. Morris, Reg. No. 46,066
WOLF, GREENFIELD & SACKS, P.C.
600 Atlantic Avenue
Boston, Massachusetts 02210
Tel. (617) 720-3500

Attorney's Docket No.: S1022.80583US00
Dated: October 13, 2005